Validation of a New Analysis Program for Law of Comparative Judgment Data

by
James J. Crile, US Army TARDEC
and
James R. McManamey, US Army NVESD

ABSTRACT

The psychophysical Law of Comparative Judgment (LCJ) has been shown to have substantial value in assessing the signature of camouflaged or signature managed (C/SM) military assets when there is a need for a psychophysical measure of effectiveness. However, statistical tools for analyzing LCJ data are not widely available, have until now only had command-line interfaces, and require extensive interaction with the analyst. This prompted the Army to produce a new computer program that combines the functionality of several older command-line programs and has a more convenient windows interface. Exercising normal caution with new computer code, it is necessary to verify the new program to ensure that it correctly performs the desired tasks. This process requires a number of test cases: examples of each different class of results to be expected. Additionally, some of the changes in the program are expected to affect the precision of values for LCJ scales. Thus, since the values obtained with the new program are expected to be different than those obtained with the old one, it is necessary to verify that the results are equivalent. This paper describes the nature of the changes, the validation process, the nature of the validation data, and the results of the validation effort. The conclusion states that the new program does give results that are very similar to those obtained with the old programs.

BACKGROUND

The Law of Comparative Judgment (LCJ) is a method of psychological scaling that was introduced by Thurstone in 1927[1] and refined thereafter. It was summarized in 1958 by Torgerson[2], a staff member of the Lincoln Laboratory at Massachusetts Institute of Technology. This was a seminal work, explaining, among other things, both the theory and methodology of calculating an LCJ scale. It was of sufficient importance that it was reprinted in 1985, although it is now out of print[3]. Without a computer of substantial capacity, calculating an LCJ scale is time consuming and error prone, but the rapid growth of computer speed and memory capacity in the 1970's removed the computational impediments to its use, with the exception of the lack of suitable software. In the period from 1993 to 1997 that impediment was removed when Copeland, et al., adopted the LCJ for their X-based Perceptual Experiment Testbed (XPET)[4]. Since that time, the LCJ has enjoyed a surge of popularity in military evaluation of target signatures[5, 6, 7].

An LCJ Scale as a Model

An LCJ scale is actually a statistical model of the raw data that it represents. The data are viewed as being a function of a single variable that describes the intensity of a psychophysical quantity. This quantity may be any perceived quality, tangible or intangible, such as loudness, beauty, or distinctness. Even if the quality has a tangible, physical component, such as loudness, the LCJ is concerned with the psychophysical perception of that quality. Since there is no way to directly measure the intensity of this internal, psychophysical quality, the subjects are simply asked to decide which of two stimuli has more of the quality being rated. The stimuli are presented either at the same time in close physical proximity to each other or in the same place in close temporal proximity to each other. By using appropriate statistical methodologies, one may process the data from a suitable assessment session and arrive at the scale values. The scale values are then predictors of the raw response data in much the same way that a regression line is a predictor of data that is a function of two variables, one dependent and one independent. For example, suppose that I fill the gas tank of my car. Then, after driving 344.1 miles, it needs 14.8 gallons of gasoline to fill it back up. I repeat this process 3 more times as shown in Table 1. The result is shown

in the graph in Figure 1. The graph also shows a trend line found by linear regression. It is a model of gasoline consumption in my car. It shows that, on the average, I am able to drive about 23.3 miles for each gallon of gasoline that I use, plus about 8.0 additional miles per tank-full. If I assume a linear relationship between miles driven and gallons used, this is the best model for fuel consumption of my car based on this information.

Table 1 -- Fuel consumed for various distances driven in a fictitious automobile.

gallons	miles
14.8	344.1
15.3	372.6
17.1	400.6
16.5	395.8

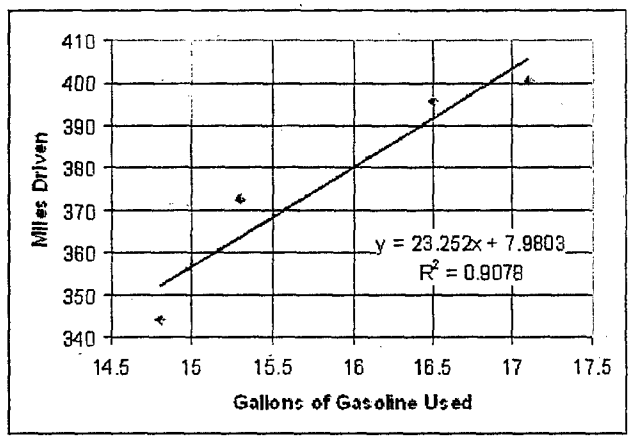

Figure 1 – Four observations of fuel consumption and associated miles driven (Table 1) along with the regression line that best describes an assumed linear relationship between the two quantities.

In the same way, the LCJ develops the linear model that best explains the responses to a set of stimuli when they are rated on some trait. The difference is that the input for calculating an LCJ scale is a two-dimensional array of responses (rating stimuli for relative values of a trait) and the output is the position of each stimulus on a linear scale for that trait. Just as with the scattered points in the gasoline consumption example, where the linear relationship is not at once easily identified, so the positions of the stimuli on a linear scale for the trait are also not easily determined. For example, suppose that 6 camouflaged targets are shown to 38 observers. They are shown in pairs. Each time a pair is shown, the observers pick the one that is the best camouflaged (hardest to see). Table 2 indicates the number of observers who pick each of the targets as best. We can easily see that A was chosen as better than E in 36 out of the 38 times when they were compared and better than F in all 38 of the cases where A and F were compared. Similarly, B was chosen as better than E in 34 out of 38 instances and better than F in 32 out of 38 instances. Thus, we know that A and B are better than E and F.

Table 2 -- Frequency with which each of 6 targets was chosen as better camouflaged when compared to each of the remaining targets.

		Chosen "Worse"					
		A	B	C	D	E	F
Chosen "Better"	A	0	23	28	33	36	38
	B	15	0	18	27	34	32
	C	10	20	0	26	33	28
	D	5	11	12	0	28	25
	E	2	4	5	10	0	17
	F	0	6	10	13	21	0

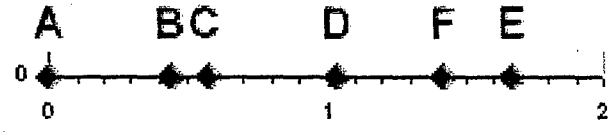

Figure 2 – Effectiveness of camouflage on 6 targets as judged by 38 observers (see Table 2). Zero is arbitrarily assigned to the most effective and higher values to less effective camouflage.

Although one can readily judge from these results that A and B are better than E and F, fine distinctions are difficult and developing a scale to know how much better any one is than any other is even more difficult. However, the statistical prescription associated with the LCJ develops a best-fit linear scale (shown in Figure 2). Using this scale, we can readily see

that treatment A is much better than treatment B. but B and C are very close to each other in effectiveness. The details of the statistical process that yields this scale are beyond the scope of this paper. For details, the reader is referred to the sources previously cited.

Need for new software

While the software included in XPET provides the means to accurately collect, combine, and process data from an LCJ evaluation session and produce a scale, there are still numerous steps. Additionally, the investigator is required to type command-line instructions in order to achieve this. Thus, to say that XPET is quick requires that one consider the older alternative of pencil, paper, statistical tables, and desk calculator. Also, practical limitations associated with field evaluations often result in an incomplete scale. Incomplete scales represent the most troubling limitation associated with the XPET software. We shall elucidate.

An incomplete matrix results whenever a cell that is not on the main diagonal contains a zero. This can occur for any of a number of reasons. A full discussion of the issue is beyond the scope of this paper, but if there are too many such cells, it also results in an incomplete scale. This means that it is not possible to produce a single scale and position all of the stimuli on it. When the scale is incomplete, one must take care that a division by zero does not occur while executing the scale development algorithm. The original XPET software was written in C and compiled for Silicon Graphics IRIS platforms. Due to some combination of default compiler options, library routines, and system features, when a division by zero is attempted, the program does not crash as one would ordinarily expect. Instead the result is flagged as NAN (Not A Number) and calculation proceeds. Subsequent calculations in which such a value is an operand, similarly result in a value that is similarly flagged. In the end, the results are printed with some of the results labeled as NAN. Copeland took advantage of this feature and did not trap division by zero errors, but instead allowed the defaults to handle them. While this was expedient and sufficient for the cases that Copeland encountered, it did not prove to be satisfactory in many instances encountered later. It has been especially problematic when the program that generates the scale is recompiled for systems that do not implement this error handling methodology.

Between the problems associated with calculation of incomplete scales and the desire to improve the user interface, it became desirable to rewrite some of the LCJ software. The authors of this paper served as a software development team to produce a new analysis program, based on certain components of XPET. In particular, the new software was to accept individual observer response files and files indicating stimulus presentation order as input and produce an LCJ scale as output. A "sum matrix" that summarizes all of the responses is a necessary intermediate product of the analysis. It was determined that such a sum matrix should be an available secondary output and an alternate input. In the course of developing the new software, a continuous function approximation was substituted for the result of trapezoid-rule numerical integration employed in XPET.

The Need to Verify the New Software

The new program functionally combines the following XPET utilities:

xpet_pairs_result_2_matrix
xpet_add_matrices
xpet_pairs_lcj

Response matrices for individual observers (output of xpet_pairs_result_2_matrix) are optional. The sum matrix (output of xpet_add_matrices) may be retained along with the LCJ scale (output of xpet_pairs_lcj). Together, these last two outputs form the most compact and useful summary of an LCJ assessment, the sum matrix preserving an exact summary of the composite data and the LCJ scale being the best linear approximation. However, the creation of new software dictates that careful verification must be performed to ensure that equivalent results are obtained. In this case, equivalent results mean:

1. Given the same individual response files, the same sum matrix will be obtained, and
2. Given the same sum matrix, an equivalent LCJ scale will be obtained.

However, improvements incorporated in the new program make it likely that small differences will occur in the resulting LCJ scales. Furthermore, to say that one LCJ scale is equivalent to another does not require that they contain the same values. An LCJ scale is an interval scale[2,3]. As such, addition of a constant or multiplication by a constant does not substantially alter the scale. Thus, to have a high correlation between two scales is sufficient. The balance of this paper is concerned with the

procedures and findings whereby it was established that the new program is equivalent to the three XPET utilities it was designed to replace, or more precisely, that when the old and new programs process the same sum matrix, an equivalent scale is obtained.

VERIFICATION PLAN

Therefore, in order to verify that the new program operates as it should, we must first identify distinct modes of execution of the algorithms themselves. Once this has been accomplished, the next step is to obtain realistic decision matrices that cause such paths of the algorithm to be executed. We must then generate LCJ scales using both the new and the old programs. At this stage, the scale obtained using the new (Crile) program will be compared to the scale obtained using the old (Copeland) program. A high correlation will ensure that the programs are giving equivalent scales. Pearson's correlation coefficient will be used to indicate the degree of similarity between the scales. Then, each scale will become input to another program that generates a predicted response matrix. For each response matrix, we will compare the predicted matrix to the true decision matrix. This step will indicate which of the two scales is the best model of the original data. The objective of this comparison is to determine whether either program consistently produces a scale that is a superior model of the raw responses. The mean squared error will be used to indicate how similar two matrices are. The verification plan is illustrated in Figure 3 below.

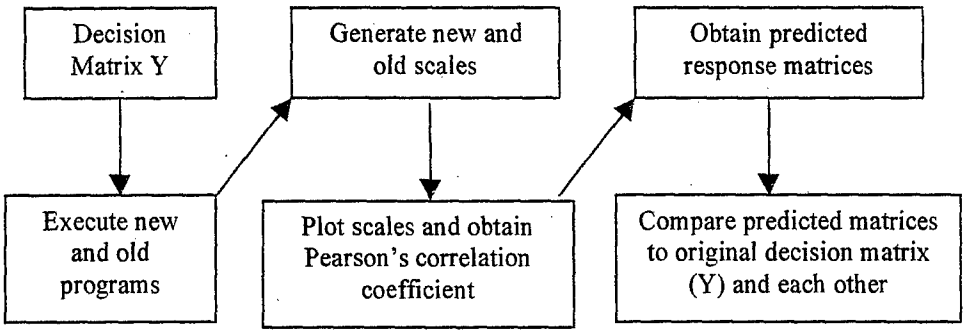

Figure 3 – The LCJ Program Validation Plan

In order to evaluate the successfulness of this verification, we will first need to establish some acceptance criteria. In general, we would say that a correlation coefficient of 0.95 between the two LCJ scales would be sufficient, given the inherent inconsistency that observers exhibit[8]. However, we would prefer to have a correlation coefficient of 0.99 or better, since purely mathematical operations account for the observed differences between the scales produced by the two programs and the observer data (program inputs) are precisely the same. For comparing the predictions of the two scales, there need be no upper limit on the sum squared error, since that is a function of the data. What we require is that this error be roughly equivalent in the two cases and that the error not be consistently higher with the new program. Using this method, we shall see if the acceptance criteria have been met. In doing so, we will determine whether or not the verification is successful. Having planned the verification process, let us now proceed to implement it.

Step 1: Identifying Distinct Cases

In this section, we will explain the four types of scales that can result from the LCJ program. At the same time, we will explain the basics of the algorithm.

The first thing we do is input a frequency matrix like Table 2 above into the LCJ program. This matrix is converted to a probability matrix (P-matrix) with the following equation for each element in the matrix:

$$P(i,j)=M(i,j)/(M(i,j)+M(j,i)), \quad (1)$$

where P(i,j) is an element of the P-matrix and where M(i,j) and M(j,i) are elements of the frequency matrix. For example, M(5,6) is 17 in Table 2, and M(6,5) is 21. Thus, P(5,6) is 0.477 and P(6,5) is 0.553.

Now, a P-matrix can be either complete or incomplete. The P-matrix is complete if it contains zeros only on the diagonal and all other entries are non-zero. If the P-matrix is incomplete, then it contains non-diagonal zero elements. At program execution, a complete P-matrix will always result in a normal scale. The frequency matrix type M1 (see Table 3) causes such an execution path. However, we will see that there are also other possibilities.

The next step in the program is to create an X-matrix, which is based on the cumulative normal standard deviate for each non-diagonal element in the P-matrix. P-matrix elements on the diagonal are ignored. Any other elements of zero or one will cause undefined values to occur in the X-matrix, since the cumulative normal standard deviate is defined only in the open interval from zero to 1 (excludes endpoints). If the X-matrix contains undefined values, then certain operations are performed on the columns to create a matrix of differences (D-matrix). The D-matrix can also have undefined values, and sometimes entire columns of it are undefined. The exact process behind this is beyond the scope of this paper, but let us be content with this explanation and continue.

From this point, identifying the other three cases is rather simple. If the P-matrix is incomplete but the D-matrix contains no undefined columns, then the matrix was of type M2. The scale resulting from such a matrix is equivalent to the scale of M1 except that it is arbitrarily chosen to begin with zero. If the P-matrix is incomplete and the D-matrix does contain undefined columns, then we end up with two or more unrelated scales called disjoint scales. Matrix type M3 contains two of these scales, and matrix type M4 contains multiple scales also of this sort. The four types are summarized in Table 3.

Table 3-- The Four Types of LCJ Results

Matrix Type	P-matrix	Scale
M1	complete	normal
M2	incomplete	normal
M3	incomplete	two disjoint scales
M4	incomplete	multiple disjoint scales

Step 2: Obtaining Exemplar Data

In Figures 4 - 7 below, we can see the four input matrices that were chosen for the validation. These matrices will yield P-matrices and scales as shown in Table 3. For example, M1 will result in a complete P-matrix and normal scale.

Step 3: Generated Scales

In this section, we will proceed to show the LCJ scale(s) that result from the four above matrices with both programs. We treat the old and new programs as "black boxes," showing the outputs without examining the method of computation. Some points will be made along the way, but for a more in-depth discussion, see **Old Model Versus New Model** (below).

In the M1 case (Figure 8), we can see that stimulus 6 received the highest rating and stimulus 4 received the lowest rating. Moreover, the lowest value is negative. There is no special significance to a negative value since these are interval scales (no natural zero) and adding any constant to every value leaves the scale, in essence, unchanged.

In the M2 case (Figure 9), we first notice that the lowest stimulus in each case is ranked zero—it is not negative as with M1. As stated above, when the P-matrix is incomplete, the first value in the scale is arbitrarily assigned zero.

We can see in Figure 10 (the M3 case) that we have two disjoint scales. This occurs because some of the stimuli are sufficiently different than the others so that the gap between two adjacent stimuli is essentially infinite (i.e. beyond measure by LCJ methods). This infinite gap can be observed in M3 (Figure 6) by the responses in the upper right and lower left quarters. We have recognized this by placing a blank line between the scales (Figure 10). The first scale contains LCJ values for 5 of the 9 stimuli. Here they are actually stimuli 1 through 5, but in general this need not be the case. The second scale

M1

0	218	230	82	166	211	197	172	220
70	0	124	37	70	179	163	137	190
58	164	0	29	85	184	168	143	195
206	251	259	0	209	160	143	117	171
122	218	203	79	0	224	211	186	232
77	109	104	128	64	0	121	104	57
91	125	120	145	77	167	0	119	59
116	151	145	171	100	184	169	0	88
68	98	93	117	56	231	229	200	0

Figure 4 – Response matrix that yields a complete P-matrix and a normal scale.

M2

0	14	14	18	14	16	12	17	20	12	13	18	19	22	17	19	14	20	22	16
8	0	12	16	12	14	10	15	19	10	11	16	17	21	15	18	12	19	22	13
8	10	0	15	11	13	9	14	19	9	10	15	17	21	14	17	11	19	22	13
4	6	7	0	7	9	5	10	16	5	6	11	13	19	10	14	7	16	21	9
8	10	11	15	0	13	9	14	19	9	10	15	17	21	14	17	11	19	22	13
6	8	9	13	9	0	7	12	18	7	8	13	15	20	13	16	9	17	21	11
10	12	13	17	13	15	0	16	20	11	12	17	18	21	16	18	13	20	22	15
5	7	8	12	8	10	6	0	17	6	7	12	14	20	11	14	8	16	21	9
2	3	3	6	3	4	2	5	0	2	2	6	8	16	5	8	3	11	19	4
10	12	13	17	13	15	11	16	20	0	12	17	18	21	16	18	13	20	22	15
9	11	12	16	12	14	10	15	20	10	0	16	18	21	15	18	12	19	22	14
4	6	7	11	7	9	5	10	16	5	6	0	13	19	10	13	7	15	21	8
3	5	5	9	5	7	4	8	14	4	4	9	0	18	8	11	5	14	20	7
0	1	1	3	1	2	1	2	6	1	1	3	4	0	2	4	1	6	15	2
5	7	8	12	8	9	6	11	17	6	7	12	14	20	0	14	8	16	21	9
3	4	5	8	5	6	3	8	14	3	4	9	11	18	8	0	5	13	20	6
8	10	11	15	11	13	9	14	19	9	10	15	17	21	14	17	0	19	22	13
2	3	3	6	3	5	2	6	11	2	3	7	8	16	6	9	3	0	19	4
0	0	0	1	0	1	0	1	3	0	0	1	2	7	1	2	0	3	0	0
6	9	9	13	9	11	7	13	18	7	8	14	15	20	13	16	9	18	22	0

Figure 5 – Response matrix that yields an incomplete P-matrix and a normal scale

M3

0	218	230	82	166	288	288	288	288
70	0	124	37	70	288	288	288	288
58	164	0	29	85	288	288	288	288
206	251	259	0	209	288	288	288	288
122	218	203	79	0	288	288	288	288
0	0	0	0	0	0	121	104	57
0	0	0	0	0	167	0	119	59
0	0	0	0	0	184	169	0	88
0	0	0	0	0	231	229	200	0

Figure 6 – Response matrix that yields an incomplete P-matrix and 2 disjoint scales.

M4

0	225	221	90	161	0	0	0	0
63	0	138	30	76	0	288	288	288
67	150	0	32	81	0	288	288	288
198	258	256	0	212	0	288	288	288
127	212	207	76	0	0	288	288	288
0	0	0	0	0	0	127	102	53
0	0	0	0	0	161	0	118	65
0	0	0	0	0	186	170	0	86
0	0	0	0	0	235	223	202	0

Figure 7 – Response matrix that yields an incomplete P-matrix and multiple disjoint scales.

contains LCJ values for the remaining 4 stimuli. The gap between the two scales is of unknown magnitude. Since it would imply that the gap had some known magnitude if we were to begin the second scale with any other value, the new program once again starts with zero. The old program is incapable of obtaining the second scale from the input matrix. With the old program, we must run it twice in order to get both scales. The first run gives us five good values (a partial scale) and four meaningless ones. After deleting from the input matrix those data that produced the first partial scale, we must run the program again to get the final four values. When this is done, the resulting 4-by-4 matrix is of type M1 and yields a scale that begins with a negative number. However, the scales are equivalent.

Matrix (M4) gives us multiple disjoint scales. In this case, when the new program is used, there are 4 scale segments. The first consists of 3 stimuli (4, 1, and 5 in that order). These are separated by a large unmeasurable gap from the lone stimulus 6. This latter stimulus is separated by another large unmeasurable gap from stimuli 3 and 2, which are very close together but come in that order. Finally come the remaining 3 stimuli (9, 8, and 7), which are separated by a large unmeasurable gap from the previous 2 stimuli. As in the M3 case, the old program is incapable of doing more than assign scale values to the first 3 stimuli. However, if a new input matrix is prepared that eliminates the data for these 3 stimuli, the old program yields a second scale segment for stimuli 3 and 2, placing them in that order. When the data associated with these stimuli are also eliminated from the input matrix, the old program yields a third segment for stimuli 9, 8, 7, and 6 in that order. These results are summarized in Figure 11.

Step 4: Predicted Response Matrices

For both models (the scale produced by the old program and the one produced by the new program), we can use the scales to predict the responses that would be obtained in an assessment session with any number of observations. We do this by finding the difference between the scale values for any two stimuli and treating that difference as the unit normal standard

New Program	Old Program
1 -> -0.35677	1 -> -0.361111
2 -> 0.206737	2 -> 0.210000
3 -> 0.160233	3 -> 0.161111
4 -> -0.409222	4 -> -0.413333
5 -> -0.341141	5 -> -0.345555
6 -> 0.398428	6 -> 0.403333
7 -> 0.255892	7 -> 0.258889
8 -> 0.0294193	8 -> 0.028889

Figure 8 — M1 scales

New Program	Old Program
1 -> 0.000000	1 -> 0.000000
2 -> 0.255073	2 -> 0.264620
3 -> 0.343295	3 -> 0.350409
4 -> 0.816674	4 -> 0.828804
5 -> 0.343295	5 -> 0.350409
6 -> 0.579156	6 -> 0.588304
7 -> 0.115517	7 -> 0.117778
8 -> 0.704121	8 -> 0.719804
9 -> 1.4107	9 -> 1.434803
10 -> 0.115517	10 -> 0.117778
11 -> 0.226845	11 -> 0.230936
12 -> 0.829811	12 -> 0.847304
13 -> 1.05578	13 -> 1.070304
14 -> 1.93126	14 -> 1.964277
15 -> 0.710206	15 -> 0.725304
16 -> 1.10289	16 -> 1.122303
17 -> 0.343295	17 -> 0.350409

Figure 9 — M2 scales

New Program	Old Program
4 -> 0.000000	4 -> 0.000000
1 -> 0.480099	1 -> 0.484000
5 -> 0.628046	5 -> 0.638000
3 -> 1.20195	3 -> 1.213999
2 -> 1.24765	2 -> 1.261999
9 -> 0.000000	9 -> -0.547000
8 -> 0.532673	8 -> -0.017000
7 -> 0.730068	7 -> 0.210000
6 -> 0.909686	6 -> 0.355000

Figure 10 — M3 scales

New Program	Old Program
4 -> 0.000000	4 -> 0.000000
1 -> 0.481710	1 -> 0.488000
5 -> 0.628046	5 -> 0.638000
6 -> 0.000000	3 -> 0.000000
	2 -> 0.060000
3 -> 0.000000	
2 -> 0.047670	9 -> -0.55000
	8 -> -0.02000
9 -> 0.000000	7 -> 0.210000

Figure 11— M4 scales

deviate for the differential preference for one of the stimuli over the other. A computer program already existed that implemented this process[9]. Given a scale, it gives a predicted response matrix that can be compared to the input matrix.

Figure 12 shows the predicted responses obtained from the two scales in Figure 8. We compare each of these to M1 (Figure 4) to obtain the mean squared error given by

$$MSE = \frac{\sum \left(m(i,j) - \hat{m}(i,j)\right)^2}{n}, \tag{2}$$

where $m(i,j)$ is an entry from the original sum matrix (Figure 4), $\hat{m}(i,j)$ is an entry in the matrix of predicted responses (Figure 12), and n is the number of entries in the matrix for which the error is being computed. The results are shown in Table 4, not only for the M1 (complete, normal) case, but also for the remaining cases. We have omitted the predicted matrices for these other cases to conserve space. However, it is important to note here that when we get two disjoint scales, as in figure 10, they do not predict the entire response matrix. For example, in the M3 case, the first partial scale with 5 values, predicts 25 entries in a 5 x 5 subset of M3 and the second partial scale predicts 16 entries in a 4 x 4 subset of M3.

PREDICTED (New Program)

0	205	201	138	146	223	210	187	190
83	0	139	77	84	166	150	124	127
87	149	0	82	89	171	155	129	132
150	211	206	0	152	228	215	193	196
142	204	199	136	0	222	209	186	189
65	122	117	60	66	0	128	103	105
78	138	133	73	79	160	0	118	121
101	164	159	95	102	185	170	0	147
98	161	156	92	99	183	167	141	0

PREDICTED (Old Program)

0	206	201	138	146	224	211	188	191
82	0	138	77	83	166	150	123	127
87	150	0	81	88	172	155	129	132
150	211	207	0	152	228	216	193	196
142	205	200	136	0	223	209	186	189
64	122	116	60	65	0	127	102	105
77	138	133	72	79	161	0	118	121
100	165	159	95	102	186	170	0	147
97	161	156	92	99	183	167	141	0

Figure 12 – M1 responses predicted by scales from new program (left) and old program (right).

Thus, we can predict only 41 of the 81 values in M3. These are the values that produce the MSE shown in Table 4. The remaining 40 entries in M3 provided insufficient information to generate a scale and the scale contains insufficient information to predict the responses.

Step 5: Predicted Versus Actual

Using the mean squared error to compare the input matrix with each predicted matrix, we get the data in Table 4. We can see here that the scales from M2 and M4 produce predicted matrices that are very similar to the input matrix, but M3 gives larger MSEs and those for M1 are very large. The reason for this is simply that the scale is truly a best linear fit to the data. If the input matrix is very scattered, the scale will not be able to fit very well, and thus the scale cannot accurately predict the input data.

Table 4 – Mean Squared Error of the Old and New Programs Compared to the Input Matrix

Matrix	New Program Mean Squared Error	Old Program Mean Squared Error
M1	1254.5	1253.7
M2	0.085	0.065
M3	33.7	31.3
M4	0.22	0.32*

*If only 18 values are used for which the new program generated comparables, value is 0.22

Step 6: Old Model Versus New Model

In comparing the old model to the new model, let us use three factors: (1) the R-squared error, that is, the square of the Pearson correlation coefficient, (2) the mean squared error as presented previously in Table 4, and (3) the data comparing the elements of the predicted response matrices of the old and new programs.

R-Squared Comparison

First, by computing a regression line and R-squared as in Figure 13, we can see how closely the scale values of the old and new programs correlate. This was done for each of the scales in figures 8-11.

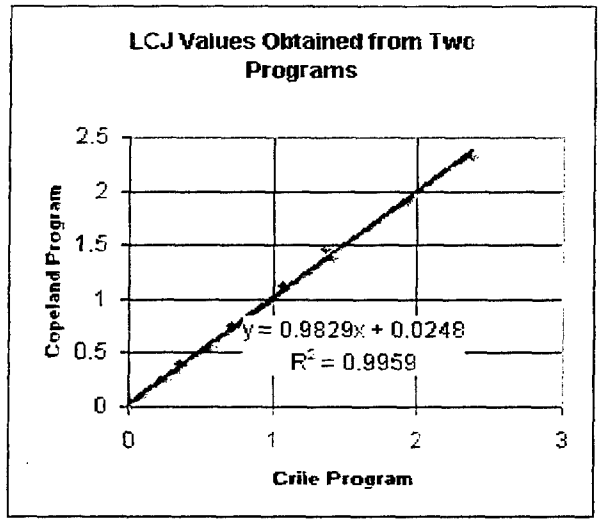

Figure 13 — Trend line for M2

The R-squared values are all very high: 1.0000, 0.9959, 1.0000, and 1.0000 respectively. In each case, Pearson's correlation coefficint is the square root of this value, with 1 indicating perfect correlation. This tells us that the two programs indeed are calculating equivalent scales, and the results they give are almost identical.

Mean Squared Error Comparison

In Table 4 above, it can be seen that the old program generally yields better (lower) mean square error. It performed better in 3 out of 4 cases.

Element Comparison

Comparing each element of each predicted response matrix to the original matrix to see which predicted response matrix had the greater error, we get the data in Table 5. Comparing the predicted matrices in this manner, we get a slightly different picture. Although the old program remains more accurate for M1, M2, and M3, we do see that for some elements, the new program was more accurate. Likewise, we get a different picture for M4 where there are the same number of entries for which each scale was more accurate.

Table 5 — Comparison of Individual Elements in Response Matrices

Matrix	New More Accurate	Old More Accurate
M1	12	18
M2	2	10
M3	10	16
M4	4	4

CONCLUSIONS AND RECOMMENDATIONS

While the new program does not yield exactly the same scale as the old one, these are "interval scales" and the same values are not required for equivalence. What is more important is the correlation between the scales produced by the two programs. After examining a number of cases that are believed to span the possibilities to be encountered, it has been

determined that there is an extremely high correlation between the scales produced by the two programs, with Pearson's correlation coefficient greater than 0.997 in all observed cases. However, comparing predicted responses from scales generated by the two programs appears to show that scales generated by the old program yield slightly better predictions.

The computational benefit expected from utilizing a continuous approximation for the area under the normal curve, in place of trapezoid-rule integration, did not materialize and may have actually added a systematic bias to the results. If such a bias exists, it is small.

In the multiple disjoint example, the old program produces no scale at all for 6 of the 9 stimuli. The improvement in the new program produces 4 scale segments, accounting for all of the stimuli. However, the results obtained are different than those obtained with the old program by the process described above. Which of these is the best representation of the data? At this point in time, the answer to this question is not clear. The new program considers all of the data available when placing stimulus 6 by itself between stimuli 5 and 3 and placing the stimuli in the order (4, 1, 5, 6, 3, 2, 9, 8, 7). Only the data relative to the 4 stimuli {6, 7, 8, 9} are considered when the old program places these in the order (9, 8, 7, 6) and provides a complete normal scale for them. If the new program is run on the same data, it produces an equivalent result. Further investigation is warranted in this matter.

With data such as that in Table 1 and Figure 1, Pearson's correlation coefficient is widely used and understood. However, there is no such metric that is known to the authors and implemented in software for describing how well an LCJ scale fits its underlying data. If there were one, it could be used to check the LCJ scales produced by the programs to determine whether the differences in the scales produced by the two programs were consistent and significant. In lieu of such a metric, in this paper we have used the mean squared error. It would be a good idea to investigate this issue further.

Finally, it is still necessary to test the part of this program that accepts the raw response data from the individual subjects and produces the sum matrix.

It would appear that the new program does calculate LCJ scales that are equivalent to those produced by the old program and incorporates some additional useful features. However, the algorithm of the old program seems as though it might be superior to that of the new one. Further investigation is warranted to determine whether there is indeed a bias in favor of the old program and, if there is, whether it is possible to change the algorithm of the new program to repair the defect. In the meanwhile, the new program could be used cautiously in place of the old one, being especially suspicious of multiple disjoint scales.

REFERENCES

[1] L. L. Thurstone, "A Law of Comparative Judgment," *Psychological Review*, 34:273-286 (1927).
[2] W. S. Torgerson, *Theory and Methods of Scaling*, John Wiley and Sons, New York (1958).
[3] W. S. Torgerson, *Theory and Methods of Scaling*, Krieger Publishing, Malabar, Florida (1985).
[4] A.C. Copeland, M.M. Trivedi, and G. Ravichandran, *Developing a Quantitative Basis for Synthesis, Analysis, and Assessment of Complex Camouflage Patterns*, U.S. Army Contract DAAK70-93-C-0037.
[5] E. L. Jacobs, "Modeling of Camouflage Screens Using Xpatch", *Proceedings of the Seventh Annual Ground Target Modeling and Validation Conference* (1996).
[6] J. R. McManamey, "Applying the Law of Comparative Judgement to Target Signature Evaluation", *RTO Meeting Proceedings 45, Search and Target Acquisition*, North Atlantic Treaty Organization RTO-MP-45 AC/323(SCI)TP/19 (2000).
[7] A. C. Copeland and M. M. Trivedi, Computational Models for Search Discrimination, Optical Engineering, 40:1885-1895 (2001).
[8] J.R. McManamey, "An Investigation of the Reliability of Search Statistics Based on Results from Paired Images," *Proceedings of the Thirteenth Annual Ground Target Modeling and Validation Conference*, Signature Research, Houghton (2002), In Press.
[9] J.R.McManamey, "LCJ2MAT.EXE" Computer program version 12/29/98, US Army, NVESD, Ft. Belvoir, VA (1998).